FACULTAT D'INFORMÀTICA DE BARCELONA (FIB)

UNIVERSITAT POLITÈCNICA DE CATALUNYA (UPC)

Grau en Enginyeria Informàtica (GEI)

Enginyeria del Software

Aplicació web per a la monitorització de vulnerabilitats de productes software

Memòria del projecte

Autor: Víctor González Garrido

Data: 18/01/2018

supervisat per

Director: Antonio Rodríguez Garcia (inLabFIB - esCERT)
Ponent: Manel Medina Llinas (AC)

Resum

Guaita és un projecte desenvolupat en l'inLab FIB UPC que pretén crear una aplicació web que té com a objectiu principal mantenir informats als Administradors de Sistemes sobre les noves vulnerabilitats que sorgeixen en els sistemes operatius, software, dispositius, etcètera amb el fi d'ajudar-los a prevenir les intrusions en els seus sistemes informàtics. Això ho aconsegueix monitorant els productes software d'una organització i mantenint una base de dades de vulnerabilitats actualitzada diàriament. A més, tots els avisos publicats en l'eina permeten conèixer a fons cada vulnerabilitat, l'impacte que pot tenir la seva explotació així com les mesures a prendre per actualitzar els sistemes afectats.

Com que l'aplicació conté informació crítica de diferents entitats, durant el seu desenvolupament s'ha donat molta importància a la seguretat de l'eina i s'ha assegurat que cada usuari només pot veure la informació que li pertoca depenen del seu rol i unitat.

Resumen

Guaita es un proyecto desarrollado en inLab FIB UPC que pretende crear una aplicación web que tiene como objetivo principal mantener informados a los Administradores de Sistemas sobre las nuevas vulnerabilidades que surgen en los sistemas operativos, software, dispositivos, etcétera con el fin de ayudarlos a prevenir las intrusiones en sus sistemas informáticos. Esto lo consigue monitorizando los productos software de una organización y manteniendo una base de datos de vulnerabilidades actualizada diariamente. Además, todos los avisos publicados en la herramienta permiten conocer a fondo cada vulnerabilidad, el impacto que puede tener su explotación así como las medidas a tener en cuenta para actualizar los sistemas afectados.

Debido a que la aplicación contiene información crítica de diferentes entidades, durante su desarrollo se ha dado mucha importancia a la seguridad de la herramienta y se ha asegurado que cada usuario sólo pueda ver lo que le corresponde dependiendo de su rol y unidad.

Abstract

Guaita is a project developed in inLab FIB UPC that aims to create a web application whose main objective is to keep System Administrators informed about the new vulnerabilities that arise in operating systems, software, devices, etc. in order to help them prevent the intrusions in their computer systems. This is achieved by monitoring the software products of an organization and maintaining a database of vulnerabilities updated daily. In addition, all the notices published in the tool allow to know in depth each vulnerability, the impact that can have his exploitation as well as the measures to take into account to update the affected systems.

Because the application contains critical information from different entities, during its development, much importance has been given to the security of the tool and it has ensured that each user can only see what is appropied depending on their role and unit.

Agraïments

Primer de tot m'agradaria agrair a Antònia Gómez i especialment a Antonio Rodríguez, per confiar en la meva primera etapa professional i transmetre'm els seus coneixements.

També m'agradaria agrair a Manel Medina per acceptar i guiar aquest TFG.

Finalment, agrair a la meva família, amics i companys tot el suport donat durant aquests mesos.

Índex

Índex de figures

Índex de taules

Introducció

Aquest és un treball de Fi de Grau per a la Facultat d'Informàtica de Barcelona (UPC). Ha estat realitzat en la modalitat B a l'inLab FIB UPC en l'àrea de seguretat coneguda com a esCERT, i està dirigit per Antonio Rodríguez i supervisat per Manel Medina.

Aquest treball consisteix en la realització d'una aplicació web que permeti a una organització monitorar els productes software que tenen instal · lats i les vulnerabilitats que els hi afecten, per així poder saber com de vulnerables són i en cas que aparegui una nova vulnerabilitat reaccionar el més ràpid possible.

1.1 Context

Cada dia apareixen desenes de **vulnerabilitats** que afecten a tot tipus de productes, com pot ser en els sistemes operatius, software, dispositius, etcètera. Per això una bona pràctica per a una empresa és mirar cada dia si alguna d'aquestes vulnerabilitats afecta algun dels productes que té instal · lats, i en cas afirmatiu comprovar en què pot afectar a la **seguretat** de l'empresa i buscar una solució.

La forma més comuna de comprovar si una vulnerabilitat afecta a algun producte que es té instal · lat és mirant el llistat de vulnerabilitats **CVE** (Common Vulnerabilities and Exposures) publicat pel **NIST[1]** (National Institute of Standards and Technology). Aquesta organització publica una llista que s'actualitza cada dia on per a cada nova vulnerabilitat trobada se li assigna un identificador i es dóna informació important com una petita descripció, una puntuació de perillositat, diferents pàgines web que poden ser d'ajuda, etcètera.

Per a cada vulnerabilitat hi ha un llistat de productes que són vulnerables. Aquests productes el NIST els identifica amb un **CPE** (Common Platform Enumeration), que és un identificador únic per a cada versió d'un producte software o sistema operatiu.

Saber de memòria els CPEs que tens instal · lats és impossible, però una forma fàcil de saber-ho és utilitzant l'**NMAP**. L'NMAP és una eina gratuïta i open source i ens serveix per aconseguir l'inventari d'una xarxa, entre altres utilitats.

L'aplicació web que s'ha fet, monitora les vulnerabilitats de productes software d'una empresa.

Per a realitzar-lo utilitza aquests dos identificadors, el CPE per als productes software i sistemes operatius i els CVEs per a les vulnerabilitats. Les CVEs les aconsegueix descarregant-les del NIST cada nit i les CPEs les pot introduir de diferents formes l'usuari, una d'elles llançant des de l'aplicació un NMAP.

El director del TFG és l'Antonio Rodríguez, responsable de l'àrea de ciberseguretat d'esCERT. El ponent és en Manel Medina Llinas, director de l'esCERT i professor de la facultat. L'Antònia Gómez, que és la responsable de tota l'àrea de Sistemes, Aules i Comunicacions de l'inLab, també ha format part de les decisions sobre les funcionalitats requerides per l'aplicació.

1.1.1 Actors

Aquest projecte va dirigit a petites i mitjanes empreses externes o institucions, així com el mateix inLab. Aquestes empreses són les que no acostumen a tenir tot un departament encarregat de la seguretat o un gran pressupost per a contractar eines molt més sofisticades, però que un atac informàtic els hi pot afectar bastant negativament.

L'eina la utilitzaran els Administradors de sistemes, que són els que s'encarreguen de mantenir els servidors actualitzats. Aquests treballadors, doncs, es beneficiaran d'un servei que automatitzarà tota la gestió de les vulnerabilitats i els hi facilitarà trobar la solució en cas de ser vulnerables.

1.2 Estat de l'art

Cada cop les empreses són més conscients de l'important que és la seguretat informàtica, també cada cop ho és més la societat, sobretot arran d'atacs tan famosos com el **Wannacry**.

Per aquest motiu cada cop les empreses dediquen més recursos a la seguretat informàtica, una pràctica comuna és encarregar auditories. Segons la Wikipedia, una auditoria informàtica consisteix en:

> La Auditoría Informática es un proceso llevado a cabo por profesionales especialmente capacitados para el efecto, y que consiste en recoger, agrupar y evaluar evidencias para determinar si un Sistema de Información salvaguarda el activo empresarial, mantiene la integridad de los datos ya que esta lleva a cabo eficazmente los fines de la organización, utiliza eficientemente los recursos, cumple con las leyes y regulaciones establecidas.

Hi ha molts tipus d'auditories depenen del que es desitgi provar però aquestes no només miren si hi ha software vulnerable, sinó que com bé s'explica a la Wikipedia, es miren altres aspectes,

com per exemple, que les pàgines web no siguin vulnerables per errades de programació, si segueixen les lleis[1] o l'eficiència i eficàcia dels sistemes i recursos informàtics.

En les auditories s'utilitzen eines que faciliten la feina als auditors. L'eina que compleix una funció similar a la d'aquest projecte és l'OpenVas. OpenVas és un framework gratuït que permet fer escanejos de vulnerabilitats. Es diferencia de la nostra aplicació en el fet que aquesta eina ha d'anar escanejant periòdicament per saber que es té instal · lat i pot ser que algun producte no el detecti si està la xarxa ben configurada.

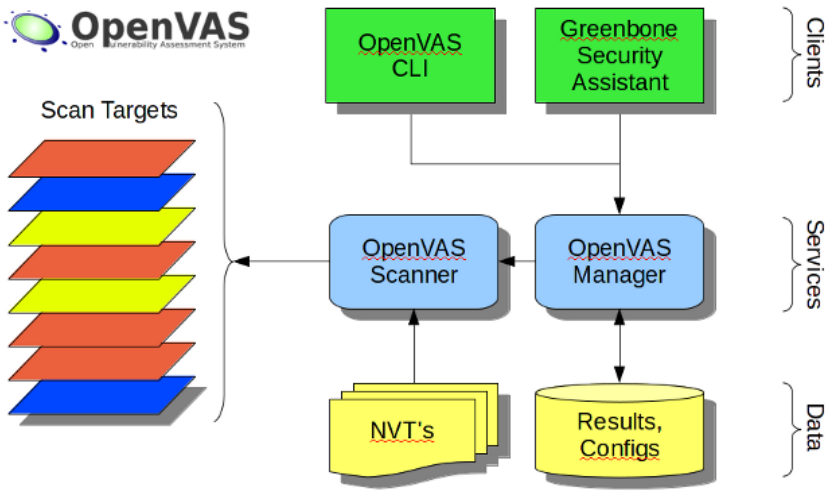

Figura 1: Estructura de l'OpenVAS 7

Per últim, per a empreses més grans s'ofereixen serveis anomenats SIEM (Informació de seguretat i gestió d'esdeveniments) que intenten detectar activitats sospitoses que poden posar en perill els sistemes d'una empresa i resoldre-les de forma immediata. Per aconseguir-lo aquests serveis són capaços de processar gran quantitat de registres i activitats sospitoses. Grans empreses com McAfee o SolarWinds ofereixen aquests serveis però a un preu molt elevat. Alguns d'aquests serveis tenen integrada una eina molt similar a **Guaita**.

La idea de fer **Guaita** és perquè a l'esCERT junt amb la UPC es va fer una aplicació amb el mateix propòsit fa uns anys (**Altair-Sigvi**), però no s'ha anat actualitzant. Per tant es va decidir fer una eina totalment nova amb la mateixa idea i solucionant tots els problemes detectats, a part de fer-la més completa i fàcil d'utilitzar, intentant automatitzar parts de l'aplicació que anteriorment s'havien d'omplir manualment.

[1] Una llei que s'ha de tenir en compte és la LOPD, que ordena com guardar la informació de caràcter personal

1.3 Formulació del problema

Sorgeixen noves vulnerabilitats cada dia i si es vol mantenir el sistema el més segur possible no val només amb fer una auditoria de tant en tant, tot i que segueix sent recomanable perquè es miren més aspectes que no només les vulnerabilitats que poden afectar.

Fent només auditories pot passar mesos entre dues auditories i en termes de seguretat això és massa. Si un cracker coneix la versió que s'està utilitzant d'un servidor Apache, per exemple, i aquest passa a ser vulnerable, és molt possible que al cap d'hores estigui penjat en algun lloc la forma de crackejar-lo, poden així, fins i tot, arribar a poder executar comandes remotament en el pitjor dels casos.

Per aquest motiu també és important estar al dia de les vulnerabilitats que afecten els sistemes operatius o software que tenim instal · lat.

El NIST publica les vulnerabilitats en format XML o JSON cosa que complica la lectura per a una persona, tot i això, si només es vol controlar un servidor, aquesta feina pot realitzar-la una persona fàcilment. En canvi gestionar més ja no és tan senzill, ja que cada servidor pot tindre un sistema operatiu diferent i diferents softwares instal · lats i això complica la revisió perquè s'ha d'aprendre tot de memòria una persona per poder comprovar si hi ha noves vulnerabilitats. A part de ser quasi impossible pot ser també una gran despesa de temps que es podria dedicar a altres coses.

Per tant, aquesta nova eina aconsegueix de forma ràpida i senzilla que els Administradors de Sistemes d'una empresa estiguin informats de les noves vulnerabilitats que sorgeixen en els seus sistemes informàtics, per així poder ajudar-los a prevenir les intrusions en els seus sistemes informàtics.

1.3.1 Objectius

Els objectius d'aquesta eina van relacionats amb facilitar i agilitzar la detecció i solució de les vulnerabilitats en un sistema informàtic, i són els següents:

- Informar de forma ràpida i senzilla les vulnerabilitats que afecten un sistema informàtic.
- Conèixer a fons cada vulnerabilitat, l'impacte que pot tenir la seva explotació així com les mesures a prendre per actualitzar els sistemes afectats.
- Automatitzar l'actualització de l'inventari.
- Generar informes i estadístiques sobre l'evolució dels tiquets i vulnerabilitats. Aquests informes es poden personalitzar i rebre'ls per correu.

1.3.2 Requisits

A continuació es llisten els requisits, tant funcionals com no funcionals, de l'aplicació web.

Funcionals

- Els usuaris han de poder registrar-se.
- Els usuaris han de poder iniciar i tancar sessió.
- Els administradors de Guaita han de poder crear unitats.
- Els administradors de Guaita han de poder afegir un administrador a una unitat.
- Els administradors de Guaita han de poder crear les xarxes de cada unitat.
- Els administradors han de poder afegir un usuari a una unitat.
- Els administradors han de poder canviar el rol dels usuaris de la unitat.
- Els administradors han de poder crear equips.
- Els administradors han de poder crear grups d'equips.
- Els administradors han de poder afegir equips a grups.
- Els usuaris han de poder veure les vulnerabilitats que té un equip.
- Els usuaris han de poder veure les vulnerabilitats que té un grup.
- Els usuaris han de poder veure les vulnerabilitats que té una unitat.
- Els usuaris han de poder veure tota la informació d'una vulnerabilitat.
- Els administradors han de poder assignar tiquets a usuaris de la unitat.
- Els usuaris han de poder gestionar els tiquets de la seva unitat.

No Funcionals

- Un usuari només ha de poder veure la informació de la seva unitat.
- Ha d'haver-hi diferents rols.

- Ha d'haver-hi un sistema de tiquets on es crea un per a cada vulnerabilitat que afecta el sistema.
- Només s'han de poder crear equips que la IP pertanyi a la xarxa de la unitat.
- El sistema ha d'incorporar l'eina NMAP.
- No s'han de poder llençar NMAPs a IPs privades.
- Ha d'haver-hi un sistema de plans.
- S'han d'encriptar les contrasenyes i IPs a la Base de Dades.
- La plataforma ha de ser robusta i mantenir la seva integritat, assegurant que no és subjecte de possibles errors interns.
- La plataforma ha de ser autocontinguda, és a dir, tota l'automatització no pot dependre del cron o eines similars.

1.4 Abast

L'abast d'aquest projecte és desenvolupar per a l'inLab una eina que cada empresa que la contracti pugui personalitzar-la i automatitzar-la al seu gust, perquè els seus Administradors l'únic que hagin de fer és que quan es creï un nou tiquet assignat a una vulnerabilitat, sàpiguen ràpidament a quin equip està i puguin trobar la solució de forma ràpida. Així, només hauran de mirar l'aplicació un cop al començament del dia, o llegir l'informe que se'ls envia per mail si han activat aquesta funcionalitat, i la resta de la jornada dedicar-la a altres funcions.

1.5 Metodologia i rigor

1.5.1 Mètodes de treball

Per desenvolupar aquesta aplicació es disposen de dos servidors i un equip d'oficina.

- S'utilitzarà una única Base de Dades (BD) en un dels servidors que emmagatzemarà tota la informació.
- S'utilitzarà l'altre servidor per a deixar l'última versió sempre activa i poder fer proves o demos.

- S'utilitzarà l'equip d'oficina per a desenvolupar l'eina, connectat a la BD del servidor.

Quan estigui la versió definitiva depenen dels usuaris que la utilitzin i la quantitat d'informació que gestioni es prendrà la decisió de si mantenir aquesta distribució o canviar a servidors més potents.

Per al backend es creu que la millor opció és utilitzar el framework de Python **Django**. És un framework dissenyat per a fer pàgines web, fàcil d'aprendre i utilitzar, i amb una documentació molt bona. A part, els seus models són molt pràctics i faciliten molt les crides a la BD. El fet de ser en Python també és un avantatge perquè a part de ser un llenguatge fàcil, hi ha moltes llibreries fetes que poden ser molt útils, com per exemple, una llibreria per fer crides NMAP de forma senzilla que inclou un parser d'NMAP a un diccionari de Python o llibreries de tractament d'IPs i Xarxes. La IDE que s'utilitzarà és **PyCharm**.

Per al frontend s'adaptarà la plantilla gratuïta **Gentelella** de colorlib. És una plantilla molt completa de Bootstrap que inclou diferents gràfiques i taules que poden ser molt útils per a representar tota la informació que es guarda.

Per a la base de dades, s'utilitzarà el SGBD (Sistema de gestió de bases de dades), el PostgreSQL.

1.5.2 Eines de seguiment

Es farà servir la metodologia àgil Scrum per al seguiment del desenvolupament de tot el projecte. Les feines pactades a les reunions s'apuntaran en forma de tasques al Trello i s'organitzaran de tal forma que les més prioritàries seran les que es realitzin en cada iteració.

La documentació es farà a la wiki del repositori git privat que té l'inLab i si calgués fer una documentació per a algú altre de l'empresa, es farà ús del Redmine privat de l'inLab.

Planificació del projecte

2.1 Planificació inicial del projecte

El projecte va començar l'1 de desembre de 2016 i s'explicava en la fase de GEP de la següent manera:

Les tasques s'agrupen en 3 fases: la planificació, el desenvolupament i la documentació. Hi ha una dependència linear entre aquestes fases, per tant, s'han de fer en aquest ordre.

Planificació

S'han de definir uns requisits bàsics abans de començar qualsevol projecte. Primer de tot definir l'abast i context del projecte i un cop fet això, planificar el temps i el pressupost necessari.

- **Abast i context**:
  Descripció: Aquest és el primer pas de qualsevol projecte. S'ha de cercar tota la informació possible relacionada amb el projecte, descriure l'estat de l'art i context i finalment definir els objectius, context i abast del projecte. En el cas d'aquest projecte, en agafar la idea d'un projecte anterior de l'empresa no es requerirà tant de temps com en altres.

  Recursos: Ordinador.

  Duració: 20 hores.

- **Planificació**.
  Descripció: En aquest pas és quan es comença a dissenyar el projecte i a definir quant de temps dedicar a cada tasca. És la part de la planificació que més temps es dedica perquè sorgeixen bastants dubtes en haver-hi diferents tecnologies que mai s'han utilitzat.

  Recursos: Ordinador, Trello, Excel.

  Duració: 40 hores.

- **Pressupost**.
  Descripció: Per a fer el pressupost s'ha de tenir en compte els recursos que s'utilitzaran, tant els humans com els econòmics.

  Recursos: Ordinador, Excel.

  Duració: 20 hores.

Desenvolupament

El desenvolupament també va per fases, primer es prepara l'entorn instal · lant tot el software necessari, a continuació s'aprenen els llenguatges de programació que s'utilitzaran i finalment es desenvolupa l'aplicació.

Per a desenvolupar aquesta eina són necessaris 2 servidors (un per a la BD i un altre per a l'aplicació), 1 equip d'oficina, Django per al Backend i una plantilla per al Fronend. Per a la gestió, un repositori Git i el Trello.

- **Preparació de l'entorn**.
  Descripció: Per a desenvolupar aquesta eina, serà necessari configurar un servidor amb la base de dades, un altre amb el Python i Django i un equip de l'oficina amb el PyCharm.

  Duració: 28 hores.

- **Aprendre llenguatges de programació**.
  Descripció: És la primera vegada que utilitzaré Django i HTML, CSS i Javascript. Per tant, abans de començar a picar codi, he de dedicar bastant de temps a aprendre a programar amb aquests llenguatges.

  Duració: 64 hores.

- **Desenvolupació**.
  Descripció: Desenvolupar tant el Backend com el Frontend. Es començarà amb el disseny dels models de Django, a continuació els controladors i finalment el front-end. Quan ja estigui l'aplicació web, es farà l'API i els scripts. Es deixa un marge de dues setmanes per a possibles desviacions.

  Duració: 708 hores.

- **Testejar**.
  Descripció: Un cop l'aplicació sigui funcional, es posa en un servidor i es testeja amb possibles usuaris per a trobar errors no detectats durant el desenvolupament, així com canviar allò que no s'entengui.

Duració: 40 hores.

- **Desplegament al servidor definitiu**.
  Descripció: Quan ja s'ha donat per vàlida l'aplicació, aquesta es desplega en un servidor.

  Duració: 20 hores.

Documentació del projecte

Per a aquest projecte hi ha dues documentacions a fer, la primera, una interna per si algú altre ha de modificar el projecte, i la segona, la documentació de la memòria i la presentació del TFG.

- **Documentació interna**.
  Descripció: La documentació s'haurà anat fent durant el desenvolupament, per tant, en aquesta fase només s'ha de completar i unificar.

  Recursos: Ordinador i Redmine.

  Duració: 20 hores.

- **Memòria i presentació del projecte**.
  Descripció: Redactar la memòria, fer la presentació i practicar-la.

  Recursos: Ordinador i LaTeX.

  Duració: 160 hores.

2.1.1 Temps estimat

Fase	**Hores**
Planificació	80
Desenvolupament	860
Documentació	180
Total	**1120**

Taula 1: Resum de les hores per fase plantejades incialment

2.1.2 Distribució temporal

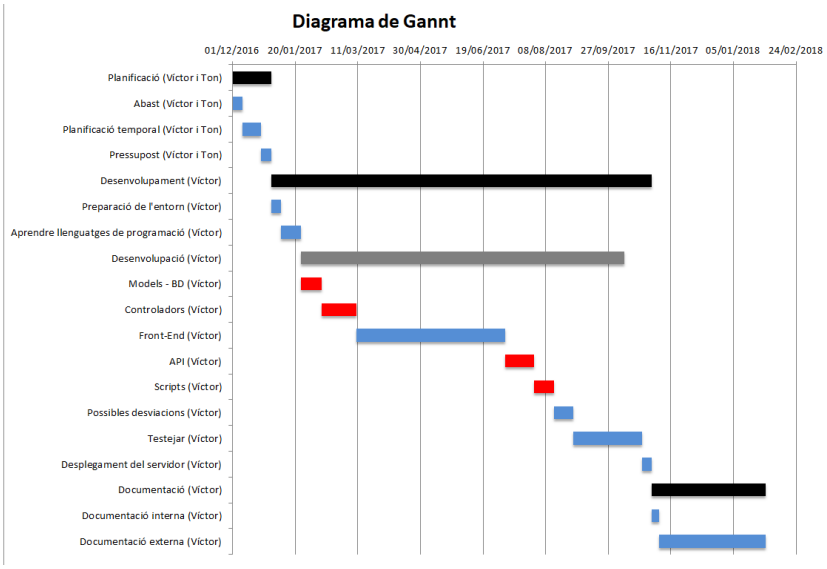

Figura 2: Diagrama de Gantt plantejat inicialment

En el diagrama es pot observar amb color negre les 3 fases principals. Dintre de cada fase, estan dividides les tasques en més petites, de color blau les que tenen menys risc i en color vermell les que tenen més.

Cada dues setmanes es farà una reunió de seguiment amb la cap de departament. Entre la planificació i el desenvolupament i el desenvolupament i la documentació també es faran reunions, però aquest cop amb més gent de l'empresa.

2.1.3 Recursos

Els recursos que s'utilitzaran en aquest projecte són els següents:

- 1 Ordinador d'oficina
- 1 Repositori de codi intern GitLab
- 1 servidor Linux per a l'execució de la plataforma en producció
- 1 servidor Linux per a la base de dades
- Software divers: Django, Python, TexStudio, PyCharm, etc.

2.1.4 Alternatives i pla d'acció

És un projecte realitzat des del principi amb metodologies àgils, per tant, les tasques es van definint a poc a poc a mesura que es va programant. Això implica que no hi ha grans tasques definides que puguin variar en el transcurs del projecte i les possibles desviacions són petites.

Tot i això, si s'hagués de canviar i fer un desviament, hi ha dues setmanes al final del desenvolupament reservades a possibles desviacions.

2.2 Planificació final del projecte

Durant la realització del projecte s'ha hagut de canviar part del plantejament inicial, però a l'utilitzar metodologies àgils aquestes desviacions no han sigut un problema per a la continuïtat del projecte. A continuació s'explicarà com ha sigut realitzat el projecte després dels canvis produïts al llarg el desenvolupament del projecte.

Les tasques es segueixen agrupant en 3 fases: la planificació, el desenvolupament i la documentació. Respecte a la planificació inicial, el desenvolupament s'ha vist allargat notablement perquè després de testejar l'aplicació es va veure que es podia tractar més eficientment les vulnerabilitats i es va decidir fer una nova versió més eficient. A causa d'aquest allargament, la fase de documentació es realitza paral · lelament amb la fase final del desenvolupament.

Planificació

La fase de planificació no ha patit cap canvi respecte el que es va plantejar inicialment.

Desenvolupament

El desenvolupament no ha patit canvis significants en l'estructura però si en el temps. L'estructura segueix sent la mateixa, primer la preparació de l'entorn, a continuació l'aprenentatge dels llenguatges de programació, el desenvolupament (amb la mateixa estructura interna) i el desplegament al servidor i testeig, aquestes dues últimes s'han invertit perquè s'ha testejat directament al servidor i no en local. En canvi, el període de desenvolupament ha sigut més llarg del previst (tot i incloent el temps dedicat a possibles desviacions de la planificació inicial) perquè s'han afegit funcionalitats que no estaven previstes inicialment i perquè durant la fase de testeig s'ha decidit implementar una nova versió més eficient.

- **Preparació de l'entorn**.
  Descripció: Aquesta fase no ha patit cap canvi respecte al plantejament inicial.

  Duració: 28 hores.

- **Aprendre llenguatges de programació**.
  Descripció: Aquesta fase tampoc ha patit cap canvi respecte al plantejament inicial.

  Duració: 64 hores.

- **Desenvolupació**.
  Descripció: El desenvolupament ha patit un increment de 272 hores respecte el plantejament inicial. Això és degut a l'increment de temps dedicat a cada una de les fases

(disseny dels models, controladors, front-end, API i scripts), però sobretot degut al desenvolupament d'una nova versió més eficient, ja que si no fos per aquesta nova versió, només s'hauria incrementat el temps en dues setmanes. En la planificació final ha desaparegut el temps reservat per a possibles desviacions que hi havia a la inicial. Duració: 980 hores.

Documentació del projecte

El gran canvi respecte a la planificació inicial, és que la documentació interna es va fent mentre es desenvolupa el projecte i per tant, no cal especificar-la en aquest apartat. Un altre canvi és el rang de temps dedicat a la documentació de la memòria, ja que la memòria la vaig començar a redactar al mateix temps que feia GEP i en la planificació inicial estava previst començar-la més tard, tot i això, el còmput d'hores estimades era correcte.

- **Memòria i presentació del projecte**.
  Descripció: Redactar la memòria, fer la presentació i practicar-la.

  Recursos: Ordinador i LaTeX.

  Duració: 160 hores.

2.2.1 Temps total

Fase	**Hores**
Planificació	80
Desenvolupament	1072
Documentació	160
Total	**1312**

Taula 2: Resum de les hores per fase definitives

2.2.2 Distribució temporal

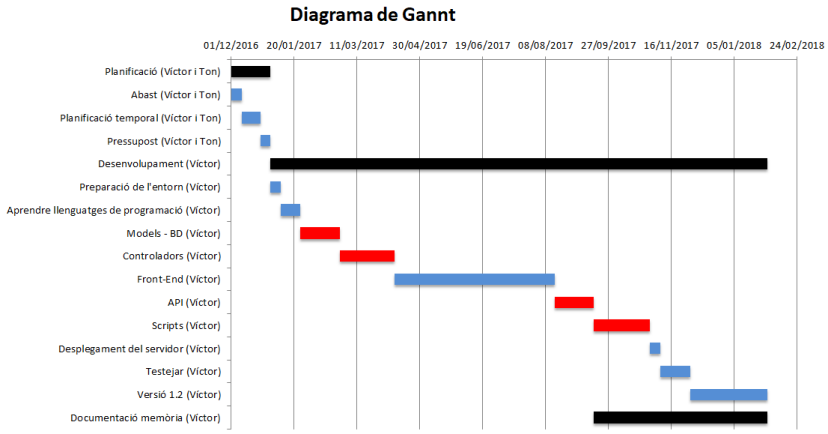

Figura 3: Diagrama de Gantt definitiu

El diagrama està representat de la mateixa forma que l'inicial. Amb color negre el total de cada fase, amb color blau les fases amb menys risc i en vermell les fases amb més risc.

Les reunions de seguiment s'han fet com estaven plantejades inicialment.

2.2.3 Recursos

Els recursos utilitzats han coincidit amb els inicials.

2.2.4 Alternatives i pla d'acció

Tot i utilitzar metodologies àgils i reservar dues setmanes per a possibles desviacions, el projecte s'ha hagut d'allargar, dues setmanes si només tenim en compte la primera versió i dos mesos i mig si es té en compte la nova versió.

2.3 Pressupost

Per a calcular el pressupost, dividirem aquest en 4 tipus de recursos i els sumarem. Els recursos són: Els humans, els hardware, els software i altres.

Com l'estimació de costos es fa a nivell d'activitats del Gantt, en la següent taula assignarem un codi a cada una de les tasques del Gantt per així ser més fàcil d'explicar els pressupostos.

Les tasques del Gantt que corresponen a programar les ajuntarem per fer més entenedor els pressupostos, aquestes tasques són: Models, Controladors, Front-End, API, Scripts i Versió 1.2.

Tasca	**Codi**
Planificació	**P**
Abast	P1
Planificació temporal	P2
Pressupost	P3
Desenvolupament	**D**
Preparació de l'entorn	D1
Aprendre llenguatges de programació	D2
Programar	D3
Testejar	D4
Desplegament del servidor	D5
Documentació	**F**

Taula 3: Codificació de les tasques del Gantt

2.3.1 Recursos humans

Aquest projecte s'ha desenvolupat en un equip de dues persones amb diferent rol dintre de l'empresa, l'estudiant i el director del projecte. El director ha tingut el rol de Project Manager i l'estudiant el de desenvolupador.

La fase de planificació la desenvoluparà pràcticament sencera el director i la resta l'estudiant, per tant, el pressupost en recursos humans és el següent:

Concepte	**Tasca**	**Hores**	**Preu/Hora**	**Total**
Project Manager	P	80 h	51 €/h	4.080 €
Software Developer	D,F	1232 h	8 €/h	9.856 €
Total				**13.936 €**

Taula 4: Pressupost de recursos humans

2.3.2 Recursos Hardware

Per a aquest projecte, ha sigut necessari un ordinador d'oficina i 2 servidors per a fer possible el projecte. Com l'empresa renova els ordinadors cada 4 anys, es tindrà en compte per als càlculs de l'amortització, que la vida útil dels ordinadors és de 4 anys i el projecte només de 14 mesos.

Cal tindre en compte, que els 2 servidors no s'utilitzen únicament en aquest projecte, sinó que també formen part del desenvolupament d'altres eines de l'empresa.

Concepte	**Tasca**	**Preu**	**Vida útil**	**Total**
Ordinador d'oficina	P,D,F	800,00 €	4 anys	233,33 €
2 Servidors	D	5.400,00 €	4 anys	1.575,00 €
Total				**1.808,33 €**

Taula 5: Pressupost de recursos Hardware

2.3.3 Recursos Software

El software que s'ha utilitzat durant el projecte és el següent:

Concepte	**Tasca**	**Preu**	**Vida útil**	**Total**
Django 1.10	D1,D2,D3,D5	N/A	N/A	0,00 €
Excel	P3	190,41	Vitalici	190,41 €
GitLab	D3,D5	N/A	N/A	0,00 €
MiKTeX 2.9	P,F2	N/A	N/A	0,00 €
OpenSUSE 12.2	D1,D5	N/A	N/A	0,00 €
Python 2.7.13	D1,D2,D3,D5	N/A	N/A	0,00 €
TeXstudio 2.12	P,F2	N/A	N/A	0,00 €
Trello	P2,D3	N/A	N/A	0,00 €
PyCharm Pro	D2,D3	N/A	N/A	0,00 € (UPC)
Windows 10 Professional	P,D,F	279,00 €	Vitalici	279,00 €
Total				**496,41 €**

Taula 6: Pressupost de recursos Software

2.3.4 Altres Recursos

Un recurs que no s'ha tingut en compte fins al moment és l'energia. Suposem que ens gastem uns 30€ al mes en l'electricitat dels servidors, de l'ordinador i la llum de l'oficina.

Concepte	**Tasca**	**Preu**	**Temps de vida útil**	**Total**
Energia	P,D,F	30,00 €/mes	N/A	420,00 €
Total				**420,00 €**

Taula 7: Pressupost d'altres recursos

2.3.5 Cost total inicial

Al pressupost inicial comptava amb dos càlculs extres, el primer eren les possibles desviacions i el segon les contingències, perquè hi havia la possibilitat que s'avariés un ordinador o servidor, teclats, ratolins, pantalles, etc. Per aquest motiu, perquè el pressupost no se superés en cap cas, es va afegir un 6 % de contingències.

Hi havia 40 hores de possibles desviacions al projecte, les quals estaven repartides entre els dos membres de l'equip:

Concepte	**Hores**	**Preu/Hora**	**Total**
Project Manager	10 h	51 €/h	510 €
Software Developer	30 h	8 €/h	240 €
Total			**750 €**

Taula 8: Pressupost de possibles desviacions

Per tant, el pressupost total, calculat amb el gantt inicial, les possibles desviacions i les contingències era el següent:

Concepte	**Cost**
Recursos humans	12.400,00 €
Recursos Hardware	1.808,33 €
Recursos Software	496,41 €
Altres recursos	420,00 €
Possibles Desviacions	750,00 €
Contingències	6% del total
Total	**16.827,74 €**

Taula 9: Pressupost calculat inicialment

2.3.6 Cost total definitiu

Com es pot observar, el pressupost final és de 172 € menys gràcies a les possibles desviacions i al 6% de contingències que s'havia afegit, ja que el pressupost en recursos humans s'ha

incrementat en 1.526 €.

Concepte	**Cost**
Recursos humans	13.936,00 €
Recursos Hardware	1.808,33 €
Recursos Software	496,41 €
Altres recursos	420,00 €
Total	**16.655,74 €**

Taula 10: Pressupost final

2.4 Sostenibilitat

2.4.1 Econòmica

Com s'ha descrit a l'apartat anterior, han sigut estimats tant els recursos humans, com els materials (software, hardware i electricitat).

Per tal que el cost del projecte fos viable s'han aprofitat les llicències de la UPC i software lliure, també les tasques s'han estimat en funció a la seva importància i se li ha dedicat un temps proporcional a cada una. En canvi, una opció per a reduir costos seria utilitzar només un servidor durant el desenvolupament, però cal remarcar, que els dos servidors també seran utilitzats en altres projectes per tal de reduir costos.

Fent referencia a les empreses (clients), mantenir treballadors alerta dels servidors, per mantenir-los segurs, és una gran despesa de diners en recursos humans, que sempre és el recurs més car. Per tant, aquesta eina pot ajudar a empreses a dedicar menys pressupost a la seguretat, però seguir sent segurs.

A més, aquesta eina també pot ser oferida com una eina extra per a les empreses que contracten a inLab (esCERT) una auditoria. Amb aquesta opció, dedicant una mica més de pressupost s'asseguren que els seus servidors segueixen segurs fins a la següent auditoria.

2.4.2 Social

La situació social a Espanya i Catalunya en aquests moments és d'incertesa, tant econòmica com social. En canvi, en el sector de la seguretat informàtica, és tot el contrari, cada cop la societat és més conscient de la seva importància i s'hi dediquen més recursos. Per tant, aquesta eina pot afavorir a empreses que estan apostant per la Ciberseguretat i fer els seus servidors més segurs enfront de possibles atacs.

Com ja s'ha comentat, hi ha una necessitat d'aquesta eina per a empreses amb més de 3 o 4 servidors, donat que estar cada dia mirant si han aparegut noves vulnerabilitats que puguin afectar els servidors pot ser molt costós en temps, i per tant, en diners.

A part, aquesta feina l'acostumen a realitzar els administradors de sistemes, que la seva funció és una altra, per tant, farà menys rutinari el seu treball i permetrà que dediquin el temps a altres feines. A més, cap col · lectiu es veurà perjudicat perquè no existeix cap ofici que sigui únicament el de mirar les vulnerabilitats d'un sistema informàtic, sinó que aquesta feina s'encarrega a un departament.

2.4.3 Ambiental

Considerant que aquest projecte és únicament la creació d'un software i que tota la documentació es farà digitalment, l'única contaminació seran les emissions de CO2 produïdes per generar l'electricitat dels servidors i ordinadors.

Per tal de reutilitzar, els servidors de desenvolupament no es compraran, sinó que s'aprofitaran els que ja disposa l'empresa. A més, un cop la vida útil d'aquests ja hagi acabat, al formar part de la UPC, els servidors i ordinadors formaran part del programa "Reutilitza"de la UPC que ofereix ordinadors reparats per a programes socials i projectes de cooperació.

2.4.4 Matriu de sostenibilitat

A continuació es mostra la matriu de sostenibilitat tenint en compte tot l'explicat en els punts anteriors.

En resum, ambientalment s'han aprofitat recursos de l'empresa però es podria haver utilitzat un únic servidor. Referent als futurs clients, milloraran la despesa d'energia, però no molt, ja que es dedica poc actualment. Econòmicament, s'han aprofitat bastant bé els recursos durant el desenvolupament, també els futurs clients estalviaran diners, en recursos humans. Finalment, en la part social, a mi personalment m'ha aportat molts coneixements interessants i, durant la vida útil, millorarà les hores de feina als administradors de sistemes, ja que faran feina menys rutinària.

	PPP	**Vida Útil**	**Riscs**
Ambiental	Consum del disseny 6 (0:10)	Petjada ecològica 3 (-10:10)	Riscs ambientals 0 (-20:0)
Econòmic	Factura 6 (0:10)	Pla de viabilitat 5(-10:10)	Riscs econòmics -5 (-20:0)
Social	Impacte personal 9 (0:10)	Impacte social 5 (-10:10)	Riscs socials 0 (-20:0)
Total	21	13	-5
	29		

Taula 11: Matriu de sostenibilitat

Desenvolupament

3.1 Disseny

El disseny es força complex i per fer més fàcil l'explicació es dividirà en tres parts. La primera, quina estructura s'ha escollit, la segona els rols de l'aplicació i finalment quines vistes web hi ha per a cada rol.

3.1.1 Estructura

L'estructura de l'aplicació ha estat pensada tenint en compte els requisits i que hi ha parts manuals i automàtiques. I un requisit important és que les parts automàtiques depenguin únicament de l'aplicació i no d'elements externs, com el cron, ja que així es facilita el desplegament.

La part manual és tot allò que fa l'usuari des del navegador, que s'explicarà en detall en el punt de vistes web.

La part automàtica són totes les funcionalitats que fan que l'aplicació estigui sempre actualitzada, com per exemple, descarregar periòdicament els CPEs i CVEs, enviar els e-mails i llançar NMAPs.

Un altre element que s'ha tingut en compte en el disseny de l'estructura és que hi ha funcionalitats que són molt complexes, com per exemple, descarregar i parsejar tots els CVEs i s'han separat de les vistes de Django per facilitar l'enteniment i manteniment del codi.

L'estructura final de l'aplicació ha quedat com mostra la figura següent:

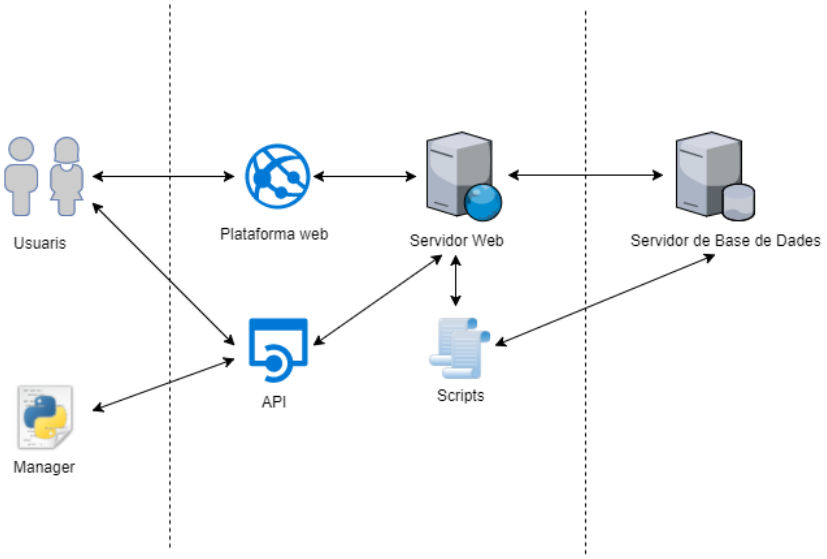

Figura 4: Estructura de la aplicació

Com es pot apreciar a la part esquerra de la figura, hi ha dues formes d'accedir a l'aplicació, els usuaris (forma manual) i el manager (forma automàtica). Els usuaris principalment accedeixen a la plataforma web i miren allà l'estat de la seva unitat, però també tenen accés a una funció de l'API on poden enviar tot el seu inventari en format JSON i s'actualitza l'estat de la seva unitat a la base de dades. El manager és l'opció que s'ha decidit perquè l'aplicació sigui automàtica i no depengui del cron. És un script fet amb Python que és llençat per l'apache i va fent crides a l'API sempre que s'hagi de fer alguna cosa. En el punt 3.2 s'explicarà més en detall la implementació.

En la part central es pot observar com en el servidor web, a part de la plataforma web i l'API, també hi ha els scripts. Aquests són cridats per les vistes de l'API i la plataforma web i són els encarregats de fer les funcions més complexes: descarregar els CVEs i CPEs, enviar e-mails, parsejar els NMAPs i actualitzar l'estat d'una unitat quan un usuari envia a l'API tot l'inventari.

Finalment, a la dreta està el servidor de Base de Dades i accedeixen tant les vistes del servidor web com els scripts.

3.1.2 Rols

L'aplicació està pensada per a oferir-la com a servei web, però també per ser utilitzada simultàniament per l'inLab. Per aquest motiu, ha sigut necessari tres rols dintre de l'aplicació. El rol d'administrador de Guaita, l'administrador d'unitat i el gestor d'unitat.

L'administrador de guaita pot fer i accedir a tot, aquest rol està pensat per administrar l'eina des de l'inLab i no per gestionar tiquets. Per tant, tindrà permisos per crear unitats, esborrar-les, afegir-li's un usuari administrador i assignar-lis un rang d'IPs o xarxes. Això últim és important, ja que els rols inferiors només podran crear equips que pertanyin a aquestes xarxes per qüestions de seguretat. Aquest rol també pot crear plans que limiten la quantitat d'equips i usuaris, entre altres aspectes, que cada unitat pot gestionar depenen del preu que paguin.

L'administrador d'una unitat està pensat perquè dirigeixi la unitat. Per tant, és l'encarregat de crear els equips i grups de la unitat, així com afegir els altres usuaris i assignar-los quin rol tindran. Un cop creada la unitat al seu gust, té algunes funcionalitats extres que els gestors no tenen, que són accés a la clau privada de l'API, l'opció de llançar NMAPs, assignar tiquets a usuaris perquè els resolguin, etc.

L'últim rol és el de gestor i està pensat per gestionar els tiquets. Per tant, aquest rol només té permisos per veure l'estat de la unitat i els tiquets i gestionar-los.

3.1.3 Vistes web

Abans de començar, cal remarcar que un usuari pot pertànyer a més d'una unitat sense cap problema. De cara a empreses externes no es pot perquè cada empresa és una unitat, però dintre de l'inLab pot passar i per això està implementat així.

Les vistes web s'han dissenyat pensant tant en els objectius com en els requisits de l'aplicació.

Per tant, la primera pantalla que es troba és la del login amb l'opció de registrar-se. El registre és obert a tothom, però al registrar-se es demana l'organització a la qual es vol unir l'usuari. Això s'ha decidit així perquè els administradors a l'hora d'afegir usuaris a la unitat, per seguretat no se'ls hi llistarà tots els de la plataforma, sinó que només els que hagin posat en el registre el nom de l'organització en qüestió.

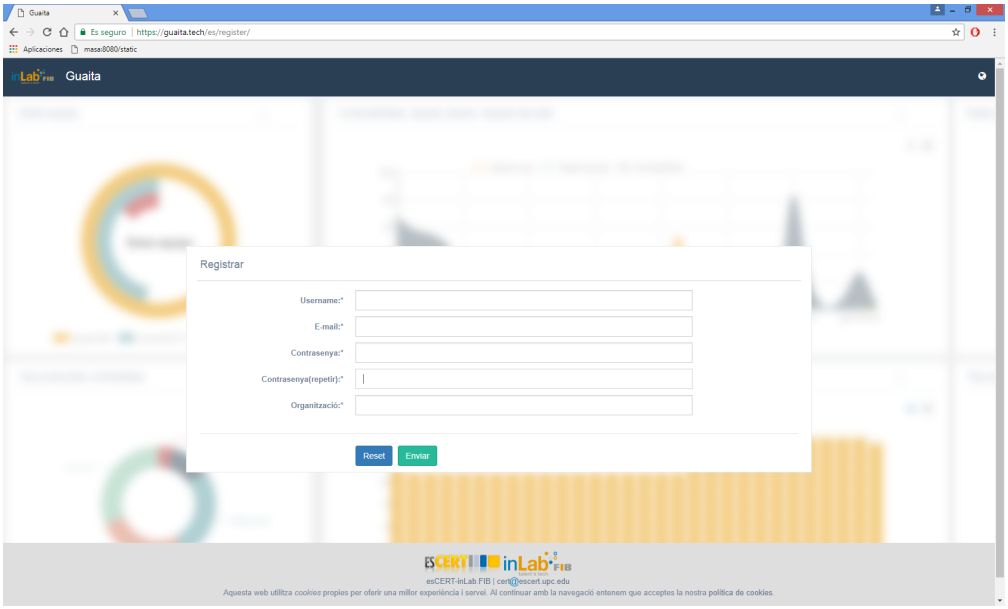

Figura 5: Captura de pantalla del registre

Un cop logejat, en totes les pantalles hi ha un menú superior on es pot veure totes les notificacions, seleccionar l'idioma (català, castellà, anglès), els NMAPs fets o pendents de fer, el perfil i la unitat o unitats que l'usuari pertany.

A la pantalla principal hi ha quatre gràfiques que permeten veure de forma senzilla i intuïtiva l'evolució dels tiquets i l'estat actual de les seves unitats (en cas que pertanyi a més d'una) o de la seva unitat (si només pertany a una).

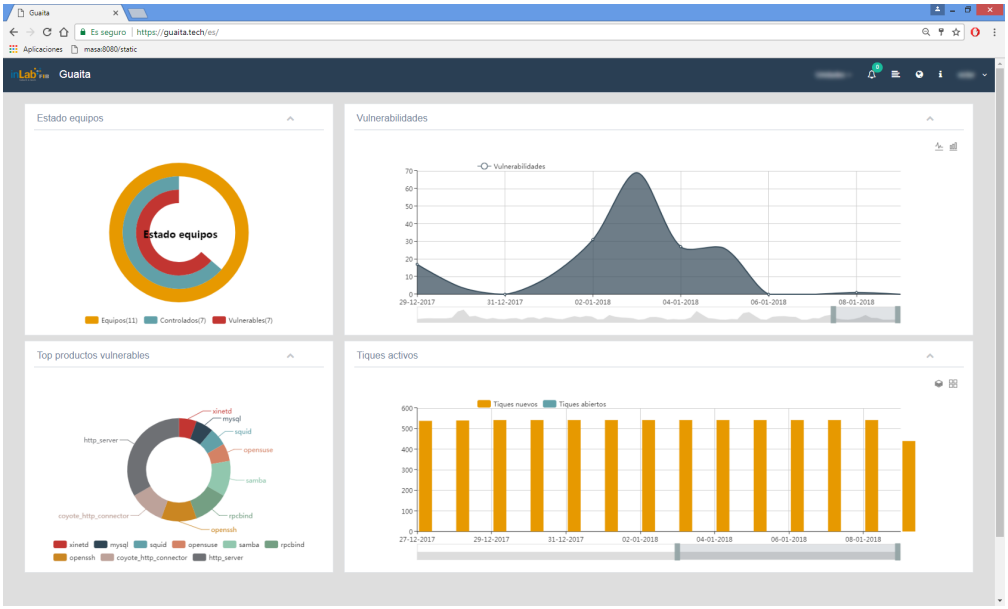

Figura 6: Captura de pantalla de la pantalla principal

Hi ha un conjunt de vistes que pretenen informar de forma ràpida i senzilla les vulnerabilitats que afecten un conjunt d'equips. Aquestes vistes són les que mostren tota la informació d'una unitat, d'un grup, d'un equip o d'un port. Estan formades per diferents pestanyes i cada pestanya conté una informació en concret. Totes comparteixen les pestanyes d'informació i vulnerabilitats, la d'informació fa un resum intuïtiu i visual de l'estat actual i la de vulnerabilitats llista totes les vulnerabilitats que afecten actualment. Hi ha altres possibles pestanyes que depenen de la vista que s'està veient i mostren informació que pot ser útil per aquella vista en concret, com per exemple, un llistat de tots els equips de la unitat o un llistat de tot el software d'un equip.

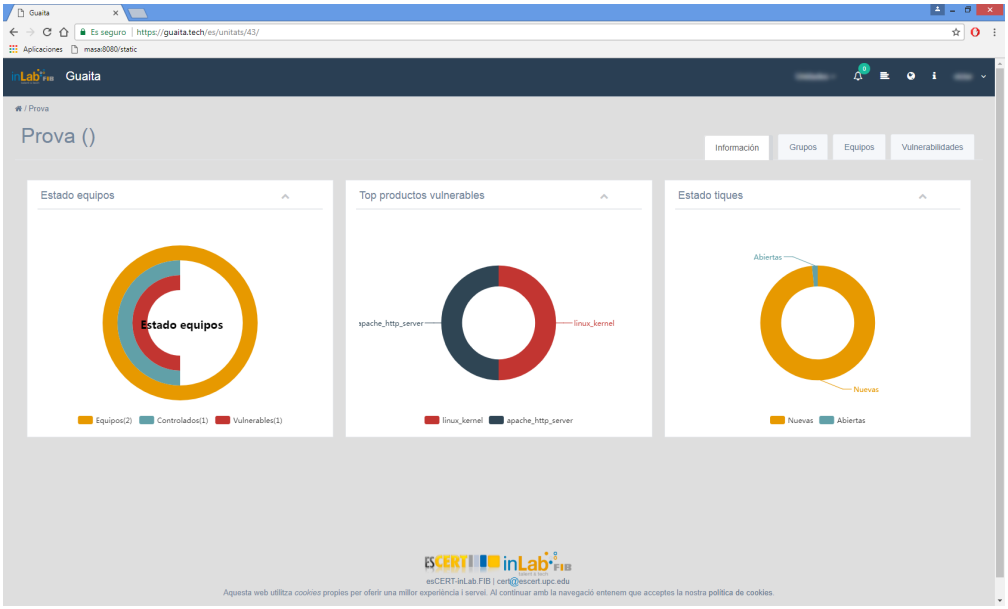

Figura 7: Captura de pantalla de la vista d'una unitat

En aquesta captura es pot apreciar ràpidament com hi ha dos equips a la unitat, un d'ells vulnerable, que els dos productes vulnerables són un Linux i un apache i que només una minoria dels tiquets actius a la unitat estan sent gestionats (tiquets oberts).

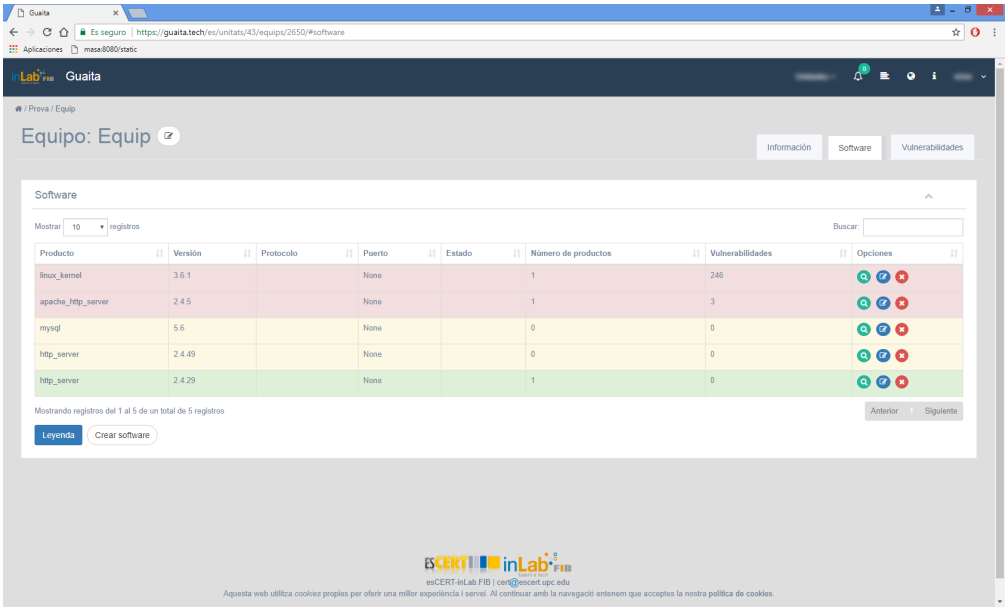

Figura 8: Captura de pantalla de la vista d'un equip

En la captura anterior es pot veure tot el software instal · lat a un equip. És important la diferència de colors, el vermell és que el software és vulnerable, el groc és que no se l'hi ha assignat encara cap CPE i l'eina és incapaç de saber si és vulnerable o no, i el verd és que té un CPE assignat però aquest no és vulnerable. Com es pot observar, és molt important assignar els CPEs als softwares, perquè els dos últims són el mateix, però un té el CPE assignat i l'altre no i l'eina els tracta diferent.

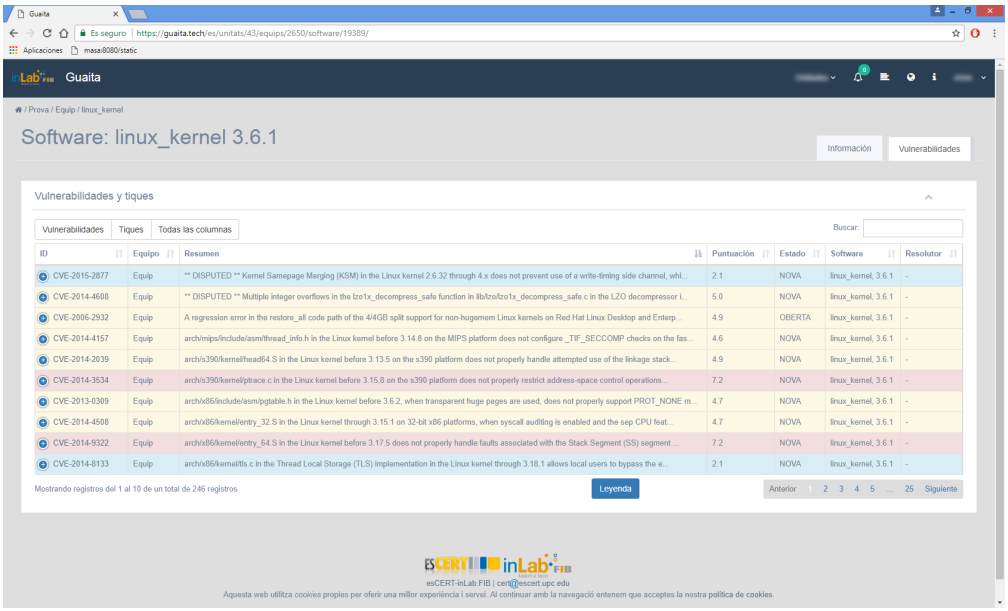

Figura 9: Captura de pantalla de la vista d'un software

La captura anterior llista totes les vulnerabilitats d'un software. Hi ha dos aspectes a comentar, el primer és que la taula pot mostrar només la informació de les vulnerabilitats, només dels estats dels tiquets associats a cada una, o com està a la captura, tota la informació junta. El segon aspecte és que també hi ha 3 colors, el blau és una vulnerabilitat de poca puntuació, la groga té una puntuació mitjana i la vermella alta.

Les vistes anteriors pretenien veure de forma senzilla tot l'inventari i mostrar informes o estadístiques que permetien veure l'evolució i l'estat actual dels tiquets. Però, l'eina també pretén donar a conèixer a l'usuari a fons cada vulnerabilitat que afecta el seu sistema. La següent vista fa aquesta funcionalitat, dóna tota la informació de l'equip i el software instal · lat, la vulnerabilitat que l'afecta i l'estat del tiquet.

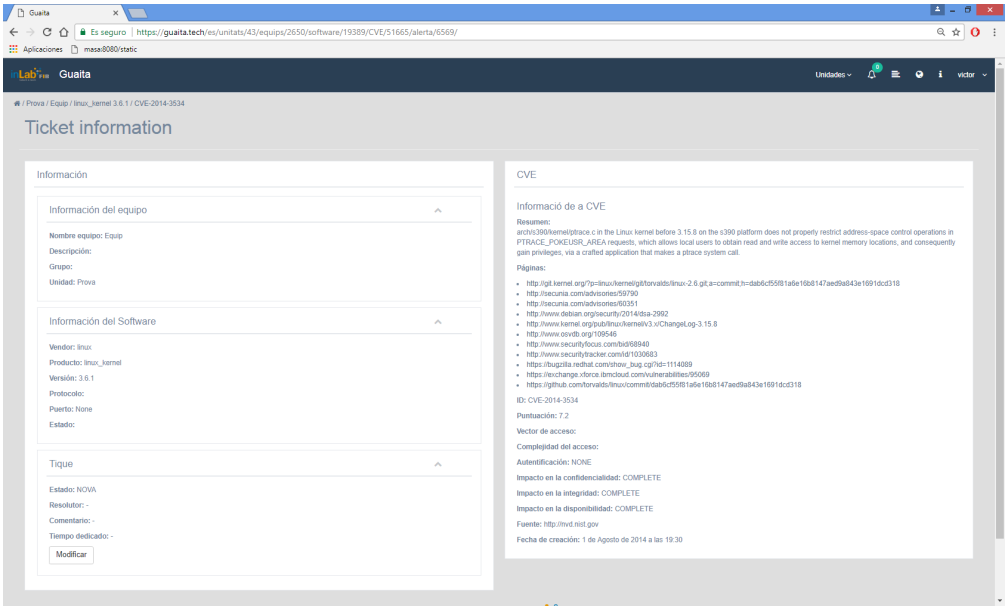

Figura 10: Captura de pantalla de la vista on hi ha la informacio d'un cve

Com es pot apreciar, a la informació de la vulnerabilitat hi ha un resum, però també informació més detallada, com el vector d'atac, la puntuació, etc. Finalment també hi ha enllaços a pàgines web on hi ha més informació, normalment relacionada amb la forma d'explotar-la i de solucionar-la.

Un cop s'ha solucionat la vulnerabilitat en el sistema, l'únic que s'ha de fer és tancar el tiquet. A continuació s'ha d'actualitzar manualment el software per la nova versió instal · lada.

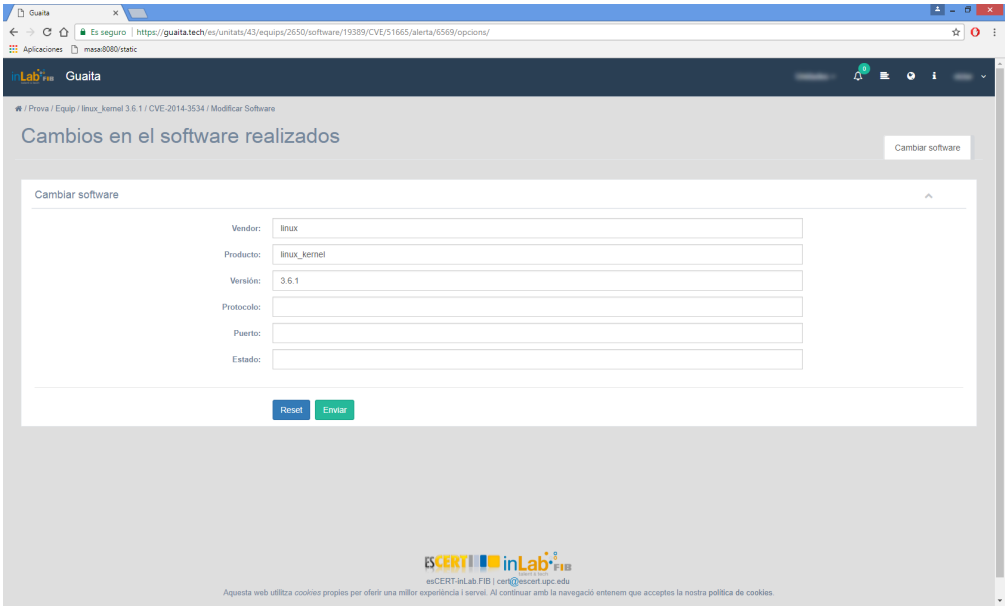

Figura 11: Captura de pantalla de la vista de la vista després de tancar un tiquet

Fins ara, eren les vistes que podia veure un usuari amb rol de gestor, a continuació es mostraran dues vistes de l'administrador d'unitat. La primera és la vista de veure unitat i la segona la d'administrar-la.

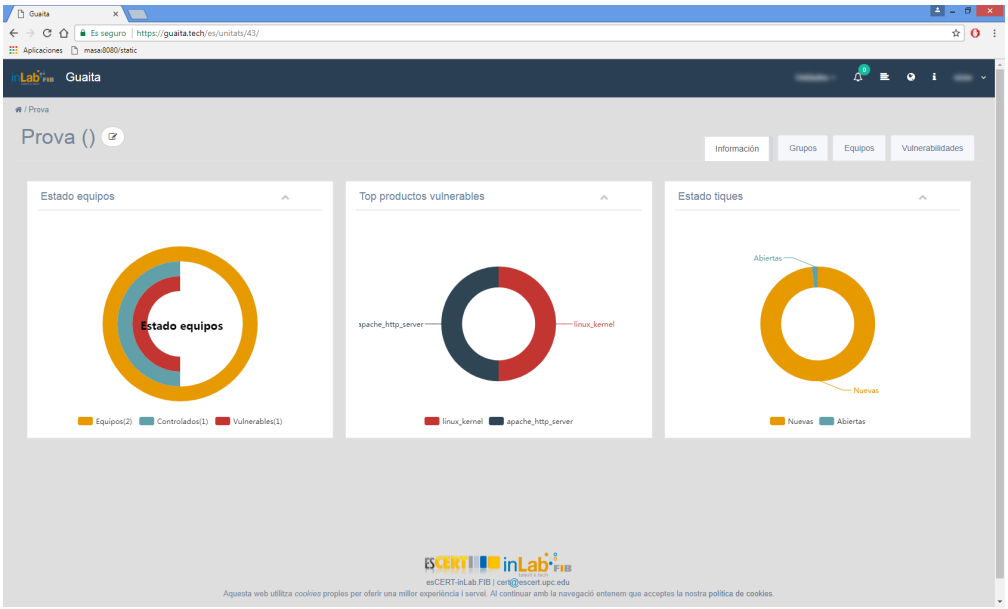

Figura 12: Captura de pantalla de la vista d'una unitat de l'administrador

La diferència entre el gestor i l'administrador en la vista superior, és que l'administrador té el botó per administrar la unitat a la part superior i més opcions, com la de crear equip, llançar NMAPs i esborrar equips.

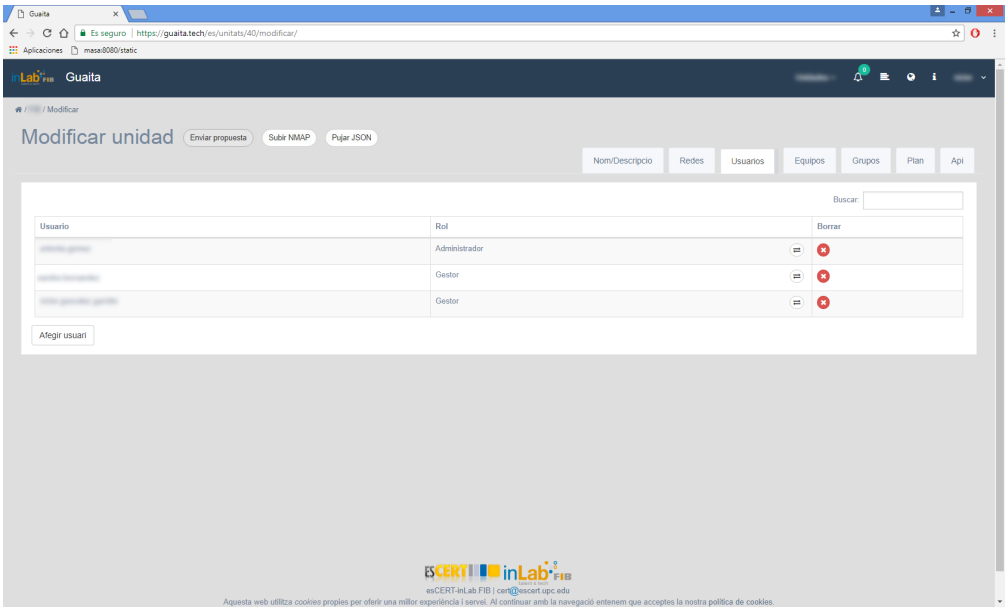

Figura 13: Captura de pantalla de la vista per administrar una unitat

Com es pot veure, és una vista molt semblant a la de veure unitat. En aquesta no es mostra la informació més general però en canvi, sí que es mostren pestanyes per a gestionar els usuaris, mirar quines xarxes s'han donat d'alta o la clau privada de l'API. També es dóna l'opció de pujar fitxers amb l'inventari en format JSON o per posar-se en contacte amb l'administrador de Guaita.

Finalment, l'última vista important, és la que permet a l'administrador de Guaita crear unitats i plans.

3.2 Implementació

A continuació s'explicarà la implementació, però el codi i certes parts de l'estructura es mantindran ofuscades per mantenir les restriccions de l'inLab. L'explicació es dividirà en 3 parts, el Backend, el Frontend i el manager que automatitza l'aplicació.

3.2.1 Backend

Com ja s'ha explicat anteriorment, per desenvolupar l'aplicació s'ha decidit utilitzar el framework web de Python, Django. Django és un framework Model-Vista-Controlador (MVC), però que Django anomena Model-Template-View (MTV). Per tant, quan es parli de vistes no són les vistes de la nomenclatura estàndard, sinó que es refereixen als controladors. A part, Django també proporciona el que anomenen aplicacions. Cada aplicació conté els seus propis fitxers de models, vistes o urls, entre d'altres. Després de saber molt per sobre l'estructura de Django, a continuació s'explicarà l'estructura del projecte.

L'estructura del backend està formada per dues aplicacions de Django, una per l'aplicació web i una altra per a l'API, així doncs, s'ha pogut separar les urls i vistes de les dues i fer el codi més net, en canvi, comparteixen els mateixos models de la Base de Dades perquè modifiquen les mateixes dades.

Un altre punt important de l'aplicació són els scripts. Com ja s'ha explicat en el disseny, els scripts són les parts de l'aplicació més complexes i per tant, s'ha decidit separar-les de les vistes per fer un codi més net i entenedor. Cada script és un fitxer Python guardat dintre de l'aplicació de Django de l'aplicació web, encara que alguns scripts també són cridats per l'aplicació de l'API.

Aplicació web

L'aplicació web està programada seguint l'estructura estàndard que recomana Django, per tant, no s'explicarà aquest apartat, però sí que s'explicarà més a fons tres aspectes importants, que són: els models, els plans i la seguretat.

Els models de Django estan formats per camps i hi ha uns quants, els més comuns, ja definits. En canvi, per al desenvolupament d'aquesta aplicació, s'ha hagut de crear i personalitzar uns quants. Sense poder entrar molt en detall, s'ha creat un camp per encriptar les IPs i uns per guardar informació força complexa que cap definit per Django s'ajustava correctament. També cal dir que els models, tot i guardar molta informació i moltes relacions (entre equips, CPEs, CVEs, unitats, grups, etc.) s'han dissenyat pensant en les funcionalitats més crítiques en temps i per això, l'aplicació és ràpida.

Un altre aspecte a comentar són els plans. Cada unitat té limitat els recursos que pot utilitzar, aquests recursos són: el nombre d'usuaris totals i administradors i la quantitat de grups i equips que poden crear. Per tal que no es vegin superats mai i donat que no només es poden crear els equips i grups manualment, sinó que també amb NMAPs i pujant arxius en format JSON, s'hi ha hagut de comprovar a cada un d'aquests llocs si ja havien arribat al màxim permès. A part, també s'ha hagut d'optar una política per a cada cas, per exemple, en pujar un arxiu JSON, es pot comprovar ràpidament al principi el nombre d'equips i si no compleix el màxim permès, no es deixa actualitzar l'estat de la unitat a la Base de Dades, en canvi, amb un NMAP no és tan senzill i s'ha optat per anar actualitzant i si en algun moment se supera el màxim, deixar de crear equips a partir d'aquell instant.

Finalment, l'últim aspecte important a comentar de l'aplicació és la seguretat, donat que es guarda informació crítica. Si es parla de seguretat, en aquesta aplicació s'ha de mirar els següents aspectes: si l'usuari està loguejat i si ho està, quin rol té i a quina unitat pertany.

Per tant, per cada vista (exceptuant unes poques com el login o el registre), abans d'accedir es comprova si l'usuari ja està loguejat i si ho està, es passa a comprovar quin rol i a quina unitat pertany. Així doncs, a cada vista només es permet accedir a uns rols en concret i si l'usuari és de la unitat. Per exemple, a la vista d'administrar la unitat amb id=1, només es permetrà l'accés, a part dels administradors de Guaita que poden accedir a tot, als usuaris amb rol d'Administrador i dintre d'aquest grup, només als quals pertanyin a la unitat amb id=1.

API

L'API s'ha implementat amb el Django Rest Framework, el framework més comú de les APIs implementades amb Django. Els models que utilitza són els creats per l'aplicació web i a l'igual que a l'aplicació, a l'API també es comprova sempre abans de crear un equip si no supera el màxim permès pel plan.

Però, l'apartat més important de l'API també és la seguretat. Com ja s'ha explicat a la figura 4, l'API pot rebre peticions del manager i dels usuaris. En el cas dels usuaris, hauran d'enviar el Token de la unitat que pertanyen i l'API automàticament detectarà quina unitat és, així doncs, només si saps el Token podràs modificar el contingut de la unitat. Com a mesura extra, les crides que només pugui fer el manager, a part, també estaran filtrades i només es podrà accedir a elles si vénen del localhost. Cal remarcar que també per motius de seguretat, com la funcionalitat de l'API és estrictament per actualitzar la Base de Dades o pel manager, només s'han implementat les funcions del manager i una vista oberta als usuaris per actualitzar la seva unitat, i no s'ha implementat cap crida que retorna informació, ja que pot ser crítica.

Scripts

Els scripts, com ja s'ha comentat, són part del codi que s'ha decidit separar de les vistes de l'aplicació web per la seva complexitat, fent així un codi més net i fàcil de mantenir. Els scripts que s'han fet són per descarregar els CPEs, per descarregar tots els CVEs, per actualitzar els CVEs, per enviar els correus, per parsejar l'estat de la unitat enviat per JSON i per parsejar un NMAP.

A continuació s'expliquen els scripts que s'encarreguen d'actualitzar les vulnerabilitats i CPEs:

L'script que descarrega els CPEs és bastant senzill, únicament descarrega del NIST el llistat de tots els CPEs en format XML, el parseja a un diccionari de Python i un per un els crea a la Base de Dades si no estan ja creats. Donat que pot haver-hi el cas que hi hagi software que no tinguin CPEs al principi, també s'ha implementat un bucle que mira tot el software instal · lat i si algun coincideix amb el CPE creat, es notifica als usuaris de la unitat que ja existeix el CPE per aquell software.

Els scripts que descarreguen tots els CVEs i el que actualitzen els CVEs són molt semblants. La diferència entre tots dos, és que el primer descarrega any per any tots els CVEs i els carrega a la Base de Dades i el segon només descarreguen les últimes vulnerabilitats creades i les guarda. Aquests scripts es descarregan les vulnerabilitats del NIST en format XML, les parsejan a un diccionari de Python i a continuació, entrada per entrada, la van guardant a la Base de Dades. Com cada vulnerabilitat té associats uns CPEs que són vulnerables, aquest script també s'encarrega de comprovar a tot l'inventari, si algun equip té instal · lat un software amb un d'aquests CPEs i crear el respectiu tiquet. Finalment, també s'aprofita aquest llistat de CPEs per comprovar si ja existeixen a la Base de Dades, i en cas contrari, també es creen.

Els scripts que s'expliquen a continuació són els que s'encarreguen d'actualitzar l'estat de les unitats automàticament. Hi ha dues formes, amb un NMAP o un JSON.

L'script que parseja l'NMAP utilitza una llibreria de Python que parseja l'NMAP a un objecte i facilita bastant la tasca. Així doncs, aquest script mira quins equips i quin software hi ha al NMAP, comprova que el nombre màxim d'equips no s'hagi excedit i crea o actualitza cada un. També, si l'NMAP ha detectat els CPEs instal · lats a cada port, mira si tenen alguna vulnerabilitat, i en cas afirmatiu, crea el respectiu tiquet. Finalment, com en el cas dels scripts de CVEs, comprova si els CPEs ja estan creats a la Base de Dades i si no els crea.

L'script que parseja JSON fa exactament el mateix que el del NMAP, però en aquest cas, al principi comprova que el format sigui el correcte, ja que no hi ha cap llibreria que ho faci.

Finalment està l'script que envia els correus.

Tenint en compte que els usuaris poden escollir la freqüència amb què se l'envia el mail i

quina informació de la unitat volen que se'ls hi mostri, l'script que envia els correus, per cada usuari registrat a la plataforma mira quines preferències ha escollit i respecte a aquestes, fa un mail a mesura. Com una de les opcions, és que el mail contingui gràfiques resumin l'estat de la unitat, l'script també s'encarrega de generar les gràfiques necessàries amb la informació guardada a la Base de Dades.

3.2.2 Frontend

Per al Frontend s'ha decidit utilitzar una plantilla que estava dissenyada per a plataformes com aquesta. La plantilla és Gentelella, produïda per Colorlib. Es va decidir utilitzar aquesta plantilla perquè coincidia amb els requisits que es tenien i perquè incluïen diferents llibreries de tercers que també eren molt útils.

Un cop descarregada la plantilla, es va decidir adaptar-la i fer-li uns canvis perquè tenia un menú lateral que no se n'anava a utilitzar i ocupava bastant lloc. Així doncs, les poques funcionalitats que anaven al menú, es van desplaçar al menú superior i es va eliminar la barra lateral. Un altre canvi rellevant en la plantilla, són els colors, que s'han adaptat al gust de l'empresa.

Finalment, les gràfiques i taules que incorpora la plantilla són molt potents i permetien fer tot el que s'havia de fer. En canvi, només amb la documentació i exemples de la plantilla no va ser suficient, i es va haver de mirar la documentació d'aquestes llibreries per poder-les aprofitar al màxim.

3.2.3 Manager

El disseny del manager és bastant senzill, simplement és un bucle infinit que va fent crides a l'API. S'ha dissenyat pensant en el fet que es pugui adaptar la freqüència de les crides, així doncs, també consta d'un fitxer de configuració a on s'especifica quan s'ha d'esperar entre crides. Paral · lelament, el manager crea un altre fitxer on va guarden les últimes hores en què s'ha realitzat cada acció per poder anar comprovant quan s'han de realitzar de nou.

El fitxer de configuració és en format JSON i se li especifica cada quants dies es descarreguen els CVEs i CPEs, cada quant s'envien els mails, a quina hora es fan les descàrregues i quan s'espera entre crides a l'API. S'ha implementat aquest fitxer perquè les accions com descarregar els CVEs i CPEs són molt costoses en temps i recursos, i així es permet a l'administrador poder decidir quan realitzar-les tenint en compte quan pot afectar menys al rendiment del servidor.

El manager és un script en Python que el primer que fa és llegir la configuració i tot seguit entra al bucle infinit. En aquest bucle es comprova quines accions s'han de realitzar. Així doncs per a cada possible acció, es demana al sistema l'hora actual, es mira quina hora s'ha

realitzat per últim cop i es comprova amb els paràmetres del fitxer de configuració si s'ha de tornar a realitzar, i en cas afirmatiu, es fa la crida a l'API, que és qui farà la descàrrega. També, per cada iteració, sempre es fa una crida a l'API per saber si hi ha algun NMAP pendent de realitzar. En cas afirmatiu, el manager aquest cop és qui fa el NMAP amb la informació retornada per l'API i li envia els resultats. Els NMAPs els realitza el manager perquè és un sistema de cues que s'ha decidit que es realitzi en background.

Finalment, fa un sleep de tants segons com s'especifiqui al fitxer de configuració i després la iteració següent. Aquest sleep també el configura l'administrador al seu gust.

Validació

4.1 Funcionament

Per dir que una aplicació funciona correctament s'han de comprovar tres criteris. Que sigui fiable, és a dir, que sempre funcioni com s'espera, que si es repeteix la mateixa acció, sempre dona el mateix resultat i que, donat que és una aplicació web, que si s'executa en diferents navegadors, en tots funciona correctament.

En aquest projecte es va decidir no fer tests, en canvi, s'ha testejat el seu funcionament des del principi manualment. Com que només era jo programant, cada cop que implementava una nova funcionalitat o actualitzava una altra, jo mateix la testejava manualment.

La forma de testejar les diferents funcionalitats és diferent, per exemple, una vista de la web on s'ha de mostrar un llistat de tots els equips d'una unitat, comprovava que si feia un select a la base de dades, el llistat era el mateix que a la web. En canvi, comprovar el funcionament d'un script és diferent, per exemple, l'script que descarrega els CVEs, havia de comprovar que els CVEs es guardaven correctament a la base de dades, però també totes les altres funcionalitats, com per exemple, que si hi havia software afectat per aquesta vulnerabilitat, es creava un tiquet o que si hi havia CPEs nous, també es creaven.

Finalment, quan ja estava la primera versió, es va decidir comprovar el funcionament utilitzant-la de forma real. En aquest punt, ja no la testejava només jo, sinó diferents persones que la utilitzaran en el futur. En aquest punt, qualsevol problema, se'm notificava i l'escrivia al Trello per a poder solucionar-lo quan fos possible.

4.2 Seguretat

En ser una eina que guarda informació sensible referent a la seguretat d'una institució o empresa, un aspecte important per validar és la seguretat. No es pot afirmar que una aplicació és 100% segura, en canvi, sí que es pot intentar que ho sigui el màxim. Una forma de validar-lo és assegurant-se que el Top 10 de la Owasp[12] no afecti la nostra aplicació. Aquest Top 10 són les 10 vulnerabilitats més comunes, per tant, si com a mínim estem

protegits d'aquests atacs, ja estem força protegits. Com el projecte ha estat desenvolupat en el departament de seguretat, el mateix departament, és qui ha fet una auditoria a l'eina un cop acabada per saber si era segura o no.

4.3 Usabilitat

La usabilitat s'ha anat validant a cada reunió que es feia i al final del projecte. Durant el transcurs del projecte, a les reunions de seguiment, s'acostumava a invitar a una altra persona de l'empresa que no formava part del desenvolupament, per comprovar si un usuari que no formava part d'aquest sabia usar l'eina. Finalment, un cop ja hi havia operativa una versió bastant completa, també es va decidir usar en un entorn real per comprovar quins problemes o millores d'usabilitat hi havia.

Obstacles

Al llarg del projecte han sorgit diferents obstacles que han afectat la seva planificació.

A continuació s'explicaran els tres més importants:

- **Canvi de plantilla:**
  Quan el frontend s'estava començant a desenvolupar s'estava fent amb una plantilla que ja s'utilitzava en altres projectes de l'Inlab, en canvi, es va decidir canviar-la per gentelella perquè era més apropiada per les característiques del projecte.

- **Versió 1.2:**
  Un cop el projecte ja s'estava provant en un entorn real, es va detectar que a vegades hi havia tiquets que eren falsos positius. Aquests existien quan la vulnerabilitat afectava un software però amb condicionants. Per exemple, una vulnerabilitat que afecta un apache, però només si aquest està instal · lat en un sistema operatiu Linux. Després d'investigar, es va detectar que el NIST proporcionava informació suficient per solucionar aquest problema i es va haver de canviar part del codi.

- **Seguretat:**
  La seguretat no ha sigut un obstacle en concret, sinó que cada cop que es dissenyava una part nova de l'aplicació, s'havia de pensar en com fer-la segura, cosa que feia més lent el desenvolupament.

Futures millores

La primera versió del projecte ja està acabada, tot i això, es podrien afegir noves funcionalitats que el farien encara més complet i funcional.

- **Ajax:**
  El frontend no està desenvolupat amb Ajax, per tant, hi ha diferents accions que afecten la usabilitat. Uns exemples són quan es creen o esborren equips, grups, software, etc. on es torna a carregar tota la vista en comptes de recarregar només la taula, o quan hi ha un NMAP escanejant de fons però la barra de seguiment no recalcula el percentatge fet fins que es recarrega la pàgina.

- **Plantilla correu:**
  El correu que s'envia als usuaris s'ha fet amb HTML però sense cap plantilla. Una forma de fer-lo més bonic seria adaptant-lo a una plantilla creada específicament per e-mails.

- **Recuperació contrasenya:**
  Fins al últim moment no es va pensar en l'opció de recuperar contrasenya (si canviar-la). Per tant, actualment si es vol recuperar, s'ha de contactar directament amb l'administrador de Guaita i que ell canviï la contrasenya manualment i se l'enviï per correu.

- **Login amb doble autenticació:**
  Actualment per accedir a l'eina només és necessari l'usuari i contrasenya. Una forma de fer més segura l'aplicació seria afegir un segon sistema d'autenticació, com per exemple, un token que s'envia al e-mail.

- **Personalitzar la puntuació de les vulnerabilitats:**
  Cada software té el camp d'estat però actualment només és informatiu per a l'administrador. Aquest camp pot ser open, filtered o closed depenen des d'on es pot accedir a aquest port, per tant, en cas que una vulnerabilitat afecti el software, si aquest està filtered, se li podria rebaixar la puntuació perquè és més difícil d'explotar-la.

Conclusions

Els objectius proposats al principi del treball (punt 1.3.1) anaven relacionats amb facilitar i agilitzar la detecció i solució de les vulnerabilitats en un sistema informàtic, i tots s'han pogut realitzar, per tant, es pot dir que el projecte ha sigut un èxit. Encara que com ja s'ha explicat al capítol d'obstacles (punt 5), hi ha hagut alguns factors que han dificultat acabar el projecte en el temps previst. Tot i això, gràcies a les metodologies àgils utilitzades, cada setmana s'anava prioritzant les tasques a fer i per això s'ha pogut finalitzar.

Per tant, si a tot això li sumem que la prova en un entorn real ha sigut positiva i que s'ha pogut millorar el tractament de les vulnerabilitats en la nova versió, també podem afirmar que aquest projecte té un gran potencial perquè permet d'una forma molt senzilla saber l'estat de tot un sistema informàtic que pot ser molt complex.

Per acabar, personalment aquest projecte també ha sigut un èxit, ja que no només he pogut aplicar diferents tècniques apreses a la universitat en un projecte real, sinó que també n'he après de moltes altres, sobretot en temes de seguretat, que també són molt importants en el desenvolupament dels projectes.

Bibliografia

[1] NIST. *National Institute of Standards and Technology* [en linia]. [Consultat: 18/01/2018]. Disponible a Internet: <https://www.nist.gov/>

[2] CVE. *Common Vulnerabilities and Exposures* [en linia]. [Consultat: 18/01/2018]. Disponible a Internet: <https://cve.mitre.org/>

[3] CPE. *Official Common Platform Enumeration (CPE) Dictionary* [en linia]. [Consultat: 18/01/2018]. Disponible a Internet: <https://nvd.nist.gov/products/cpe>

[4] NMAP *NMAP* [en linia]. [Consultat: 18/01/2018]. Disponible a Internet: <https://nmap.org/>

[5] Wikipedia. *Auditoría informática* [en linia]. [Consultat: 18/01/2018]. Disponible a Internet: <https://es.wikipedia.org/>

[6] OpenVas. *Open Vulnerability Assessment System* [en linia]. [Consultat: 18/01/2018]. Disponible a Internet: <http://openvas.org/software.html>

[7] CVE. *NVD Data Feeds* [en linia]. [Consultat: 18/01/2018]. Disponible a Internet: <https://nvd.nist.gov/vuln/data-feeds>

[8] Django. *Django documentation* [en linia]. [Consultat: 18/01/2018]. Disponible a Internet: <https://docs.djangoproject.com/en/1.11/>

[9] PyCharm. *PyCharm* [en linia]. [Consultat: 18/01/2018]. Disponible a Internet: <https://www.jetbrains.com/pycharm/>

[10] Gentelella. *Gentelella colorlib* [en linia].
[Consultat: 18/01/2018]. Disponible a Internet:
<https://colorlib.com/polygon/gentelella/>

[11] Trello. *Trello* [en linia].
[Consultat: 18/01/2018]. Disponible a Internet:
<https://trello.com/>

[12] Owasp. *Top 10-2017 Top 10* [en linia].
[Consultat: 18/01/2018]. Disponible a Internet:
<https://www.owasp.org/index.php/Top_10-2017_Top_10/>